

# Máster en Dirección de Ciberseguridad, Hacking Ético y Seguridad Ofensiva

DESARROLLO SEGURO EN PYTHON

# LECCIÓN 04

## Auditoría y Securización de Código.

# ÍNDICE

Introducción

Análisis

Objetivos

Conclusiones

# INTRODUCCIÓN

Llegados a este punto ya conocemos las principales listas de vulnerabilidades que puede tener el software y los principales controles proactivos que podemos aplicar para desarrollar un software lo más seguro posible.

El siguiente paso será conocer herramientas diseñadas para analizar y auditar código en busca de posibles errores de seguridad y conocer las vulnerabilidades del lenguaje o framework con el que estamos trabajando.

# OBJETIVOS

Al finalizar esta lección serás capaz de:

- 1 Conocer las vulnerabilidades del lenguaje con el que estamos trabajando
- 2 Aplicar herramientas para auditar y securizar código
- 3 Saber como proceder para securizar nuestro código
- 4 Auditar aplicaciones reales

## Seguridad en Proyectos Python

Un punto muy importante a tener en cuenta cuando vamos a trabajar con un lenguaje específico es analizar y estudiar sus componentes inseguros o vulnerabilidades conocidas.

En este punto vamos a hacer un inciso sobre las funciones de Python y sus frameworks.



```
usr/bin/env python
*- coding: utf-8 -*-
m cgi import FieldStorage
ort pymysql
nt("Content-Type: text/plain")
nt("Accediendo a base de datos --> ")
nt()
m_input = FieldStorage()
e = form_input["name"].value
sword = form_input["password"].value
n = pymysql.connect(host='localhost', port=3306, user='root', password='root')
sor = conn.cursor()
cursor.execute("SELECT last_visit FROM users WHERE "
               "name='{}' AND password='{}'".format(name, sword))
q:
print("Bienvenido, tu última visita fue el {}".format(
    cursor.fetchone()[0]))
e:
print("Usuario o clave incorrectos.")
sor.close()
n.close()
```

## Seguridad en Proyectos Python

- Uso de funciones Python peligrosas, como `eval()`, sin la correcta validación de las cadenas de entrada.
- Acceso al sistema de archivos.
- Los fragmentos de código SQL y JavaScript integrados en el código fuente o las plantillas, especialmente si estamos utilizando Django o Flask. En este punto, se recomienda seguir las mejores prácticas usando mecanismos de persistencia como Django ORM: <https://docs.djangoproject.com/en/3.1/topics/db/> para persistir en las plantillas de bases de datos y evitar la inyección de SQL y XSS.



```
usr/bin/env python
*- coding: utf-8 -*-
m cgi import FieldStorage
ort pymysql
nt("Content-Type: text/plain")
nt("Accediendo a base de datos --> ")
nt()
m_input = FieldStorage()
e = form_input["name"].value
sword = form_input["password"].value
n = pymysql.connect(host='localhost', port=3306, user='root', password='root')
sor = conn.cursor()
cursor.execute("SELECT last_visit FROM users WHERE "
               "name='{0}' AND password='{1}'".format(name, sword))
q:
print("Bienvenido, tu última visita fue el {}".format(
    cursor.fetchone()[0]))
e:
print("Usuario o clave incorrectos.")
sor.close()
n.close()
```



## Seguridad en Proyectos Python

- Claves API hardcodeadas en el código fuente.
- Llamadas HTTP a servicios web internos o externos.
- Flask-Security  
<https://pythonhosted.org/Flask-Security/>
- <https://docs.python.org/3.9/library/subprocess.html#security-considerations>



```
usr/bin/env python
*- coding: utf-8 -*-
m cgi import FieldStorage
ort pymysql
nt("Content-Type: text/plain")
nt("Accediendo a base de datos --> ")
nt()
m_input = FieldStorage()
e = form_input["name"].value
sword = form_input["password"].value
n = pymysql.connect(host='localhost', port=3306, user='root', password=sword)
sor = conn.cursor()
cursor.execute("SELECT last_visit FROM users WHERE "
               "name='{0}' AND password='{1}'".format(name, password))
q:
print("Bienvenido, tu última visita fue el {0}.".format(
    cursor.fetchone()[0]))
e:
print("Usuario o clave incorrectos.")
sor.close()
n.close()
```



## Seguridad en Proyectos Python

### Herramientas para auditar código

**bandit 1.7.0**

```
pip install bandit
```



**pylint 2.7.2**

```
pip install pylint
```



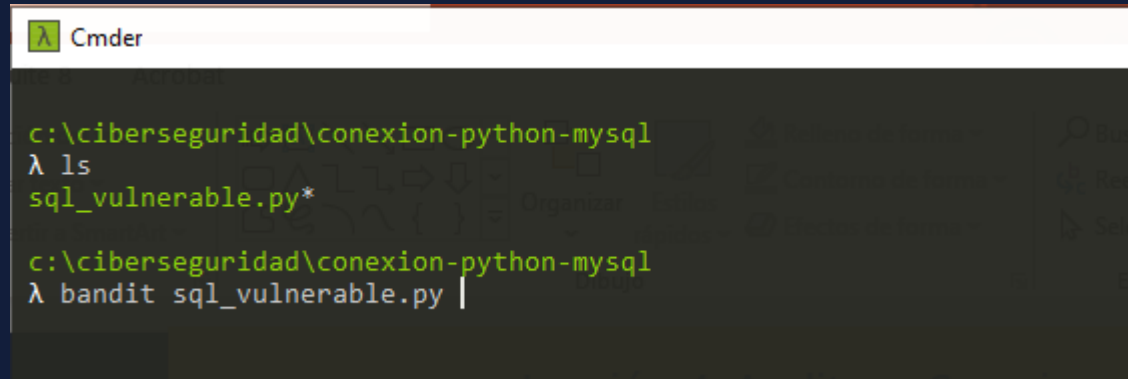
**pyflakes 2.3.0**

```
pip install pyflakes
```



## Seguridad en Proyectos Python

### Ejemplo usando bandit



```
λ Cmdr  
c:\ciberseguridad\conexion-python-mysql  
λ ls  
sql_vulnerable.py*  
c:\ciberseguridad\conexion-python-mysql  
λ bandit sql_vulnerable.py |
```

## Seguridad en Proyectos Python

### Ejemplo usando bandit

```

Test results:
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: ''
    Severity: Low    Confidence: Medium
    Location: sql_vulnerable.py:4
    More Info: https://bandit.readthedocs.io/en/latest/plugins/b106\_hardcoded\_password\_funcarg.html
3
4     db = pymysql.connect(host="localhost",user="root",passwd="",db="pokedek")
5
-----
>> Issue: [B608:hardcoded_sql_expressions] Possible SQL injection vector through string-based query construction.
    Severity: Medium    Confidence: Medium
    Location: sql_vulnerable.py:10
    More Info: https://bandit.readthedocs.io/en/latest/plugins/b608\_hardcoded\_sql\_expressions.html
9
10     cur.execute("SELECT * FROM pokemon WHERE nombre = '%s';" % pokemon)
11
-----

```

## Seguridad en Proyectos Python

### Ejemplo usando bandit

[Docs](#) » [Bandit Test Plugins](#) » [B106: hardcoded\\_password\\_funcarg](#)

[Edit on GitHub](#)

### B106: hardcoded\_password\_funcarg 🔗

```
bandit.plugins.general_hardcoded_password.hardcoded_password_funcarg(context) \[source\]
```

B106: Test for use of hard-coded password function arguments

Example:

```
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded
password: 'blerg'
  Severity: Low    Confidence: Medium
  Location: ./examples/hardcoded-passwords.py:16
15
16     doLogin(password="blerg")
```

#### 📘 See also

- [https://www.owasp.org/index.php/Use\\_of\\_hard-coded\\_password](https://www.owasp.org/index.php/Use_of_hard-coded_password)

## Seguridad en Proyectos Python

### Ejemplo usando bandit

[Docs](#) » [Bandit Test Plugins](#) » B608: hardcoded\_sql\_expressions

[Edit on GitHub](#)

## B608: hardcoded\_sql\_expressions

### B608: Test for SQL injection

An SQL injection attack consists of insertion or “injection” of a SQL query via the input data given to an application. It is a very common attack vector. This plugin test looks for strings that resemble SQL statements that are involved in some form of string building operation. For example:

Example:

```
>> Issue: Possible SQL injection vector through string-based query
construction.
  Severity: Medium   Confidence: Low
  Location: ./examples/sql_statements_without_sql_alchemy.py:4
3 query = "DELETE FROM foo WHERE id = '%s'" % identifier
4 query = "UPDATE foo SET value = 'b' WHERE id = '%s'" % identifier
5
```

#### See also

- [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)
- [https://security.openstack.org/guidelines/dg\\_parameterize-database-queries.html](https://security.openstack.org/guidelines/dg_parameterize-database-queries.html)

### SONARQUBE

#### Seguridad y Calidad del código

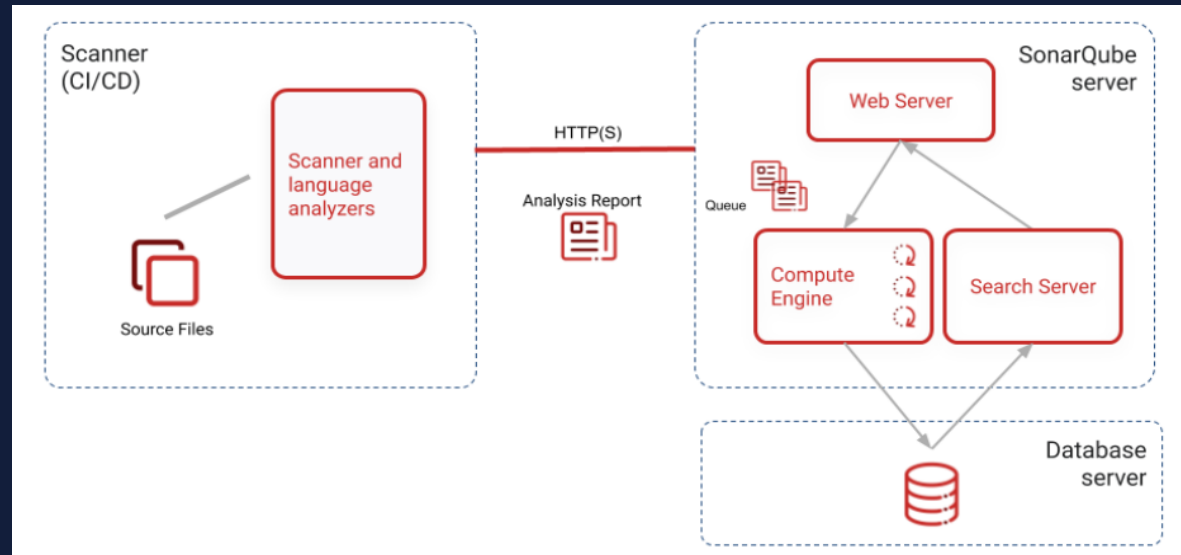
SonarQube es una plataforma para evaluar código fuente. Es software libre y usa diversas herramientas de análisis estático de código fuente como Checkstyle, PMD o FindBugs para obtener métricas que pueden ayudar a mejorar la calidad del código de nuestro software





## SONARQUBE

### Procesos SonarQube

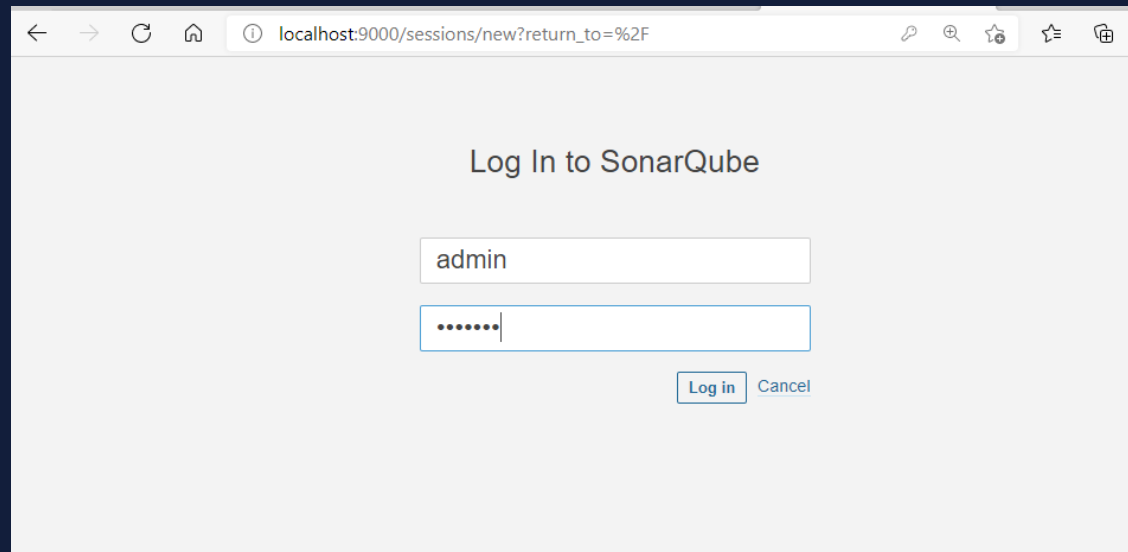


- Un servidor web que sirve a la interfaz de usuario de SonarQube.
- Un servidor de búsqueda basado en Elasticsearch.
- El motor de cálculo encargado de procesar los informes de análisis de código y guardarlos en la base de datos de SonarQube.

## SONARQUBE

### Analizando un proyecto

```
c:\ciberseguridad\sonarqube\bin\windows-x86-64  
λ StartSonar.bat |
```



A screenshot of a web browser window showing the SonarQube login page. The address bar displays 'localhost:9000/sessions/new?return\_to=%2F'. The page title is 'Log In to SonarQube'. It features two input fields: the first contains the text 'admin', and the second contains masked characters '.....'. Below these fields are two buttons: 'Log in' and 'Cancel'.

## SONARQUBE

### Analizando un proyecto

localhost:9000/projects/create?mode=manual

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

### Create a project

**Project key\*** ⓘ

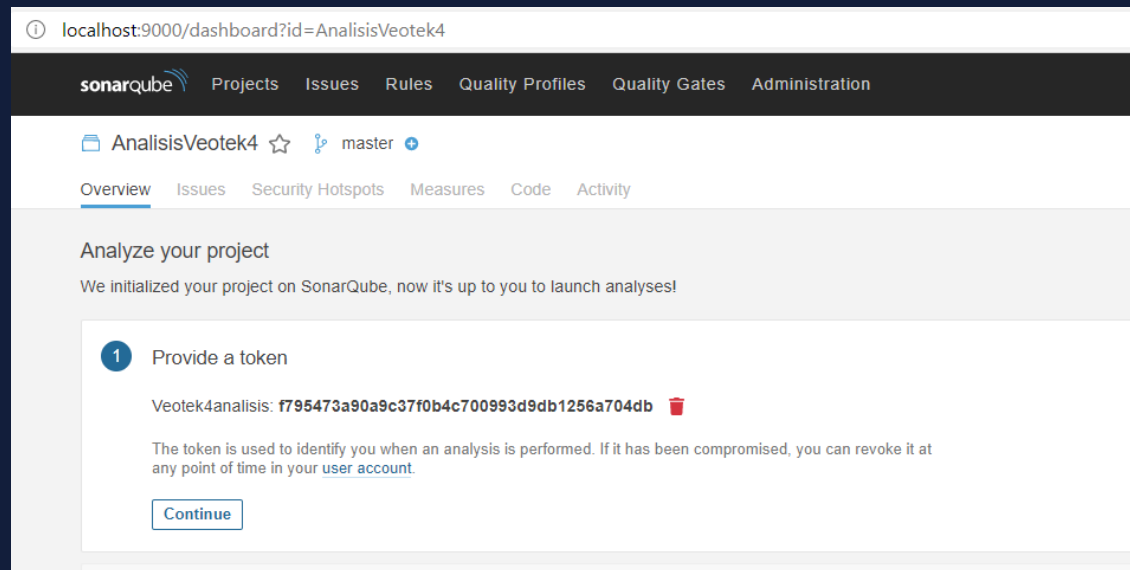
 ✓  
Up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '\_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

**Display name\*** ⓘ

 ✓  
Up to 255 characters

# SONARQUBE

## Analizando un proyecto



## SONARQUBE

### Analizando un proyecto

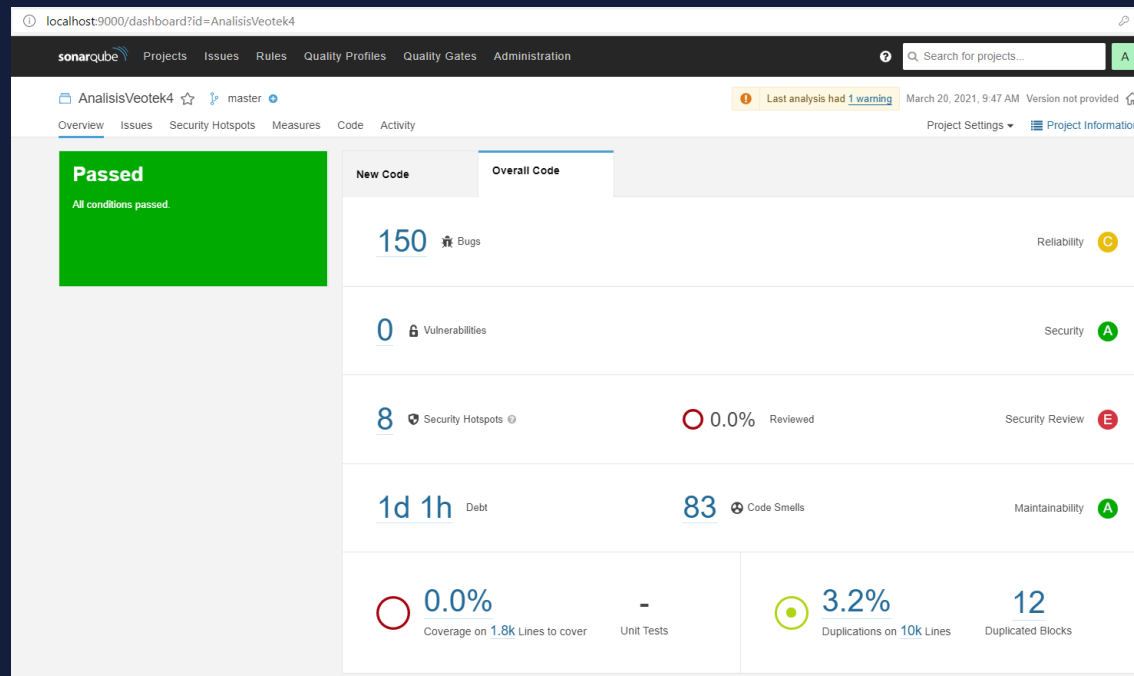
```
c:\ciberseguridad
λ cd veotek\

c:\ciberseguridad\veotek
λ ls
actividades.php      css/                 inicio.php
actualizar.php       eliminar.php         js/
add-user.php         fonts/              README.md
administrador.php    funciones.php       registrar.php
bitacora.php         horario-diario.php  reportes.php
bitacora-diaria.php  horario-mensual.php salir.php
bitacora-mensual.php horario-personal.php sesion.php
bitacora-personal.php img/               tolerancia.php
conexion.php        index.php           usuarios.php
crear-usuario.php   info.php           veotek3.sql

c:\ciberseguridad\veotek
λ sonar-scanner.bat -D"sonar.projectKey=AnalisisVeotek4" -D"sonar.sources=." -D"sonar.host.url
=http://localhost:9000" -D"sonar.login=f795473a90a9c37f0b4c700993d9db1256a704db"
```

# SONARQUBE

## Analizando un proyecto





### SONARQUBE

#### Analizando un proyecto

En los reportes hay que tener en cuenta:

- Bugs: errores de programación
- Vulnerabilidades: son errores que afectan a la seguridad.
- Hotspots: fragmentos de código relacionados con la seguridad, que deben ser manualmente revisados. Permiten solucionar problemas de seguridad y nos ayudan a aprender sobre seguridad
- Code Smells: malas prácticas de programación que dificultan que el código pueda mantenerse

En relación a calidad del código también es importante tener en cuenta el porcentaje de código duplicado.

## SONARQUBE

### Analizando un proyecto

The screenshot displays the SonarQube web interface for a project named 'AnalysisVeotek4'. The main navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. A search bar is present on the right. The project page shows a status of 'To review' and a warning that the last analysis had 1 warning. The 'Security Hotspots' tab is active, showing 8 hotspots to review. The left sidebar lists categories: Authentication (2), Weak Cryptography (1), and Others (5). The main content area shows a specific hotspot for a hardcoded credential in the file '/conexion.php'. The code snippet is as follows:

```
1 <?php
2 $conexion = mysql_connect("localhost", "root", "veotek");
3 mysql_select_db("veotek3", $conexion);
4 ?>
```

The hotspot is categorized as 'Authentication' with a 'HIGH' review priority. It is currently 'Not assigned' and has a status of 'To review'. A message indicates that this hotspot needs to be reviewed to assess whether the code poses a risk. The right sidebar provides a detailed explanation of the risk, stating that hard-coded credentials are a security vulnerability and should be stored in a configuration file, database, or management service for secrets. It also lists CVE-2019-13466 and CVE-2018-15389 as related vulnerabilities.



## CONCLUSIONES

1

Conocer las vulnerabilidades del lenguaje o framework con el que estamos trabajando nos ayuda a securizar nuestro software.

2

El uso de las herramientas de auditoría y calidad del código es muy importante en el proceso de securización del software.

3

Las herramientas de auditoría se deben combinar con los requisitos del software, legislación vigente y un ciclo de desarrollo de software seguro para que sean efectivas.

MUCHAS GRACIAS POR SU ATENCIÓN



[dtinedo@grupomainjobs.com](mailto:dtinedo@grupomainjobs.com)



Diego Tinedo Rodríguez

[linkedin.com/in/diego-tinedo](https://www.linkedin.com/in/diego-tinedo)



[twitter.com/eiposgrados](https://twitter.com/eiposgrados)



[facebook.com/eiposgrados](https://facebook.com/eiposgrados)



[instagram.com/eiposgrados](https://instagram.com/eiposgrados)